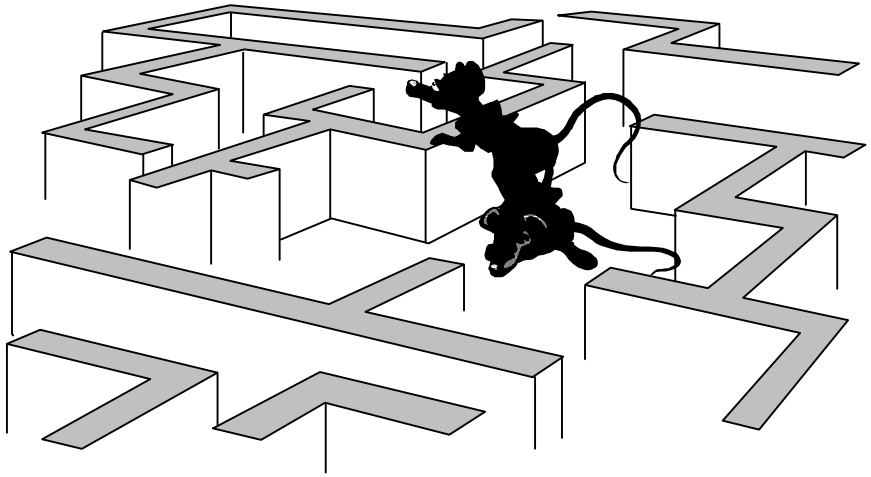
Einleitung zur Vorlesung logische und funktionale Programmierung



Was uns erwartet und

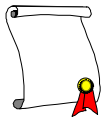
was von uns erwartet wird?!

Inhalt der Vorlesung

- Einleitung - Programmierphilosophie
- Logische Programmierung mit Prolog
- Funktionale Programmierung mit ML
- *Logisch-funktionale Programmierung*

Organisation

- Vorlesung: freitags 9.15 - 10.45 Uhr
- in ungeraden Wochen: anschließend Übung

  Abgabetermin: freitags 12 Uhr in geraden Wochen
- Scheinerwerb 
  - 70% der Aufgaben bearbeitet, 50% der Punkte erreicht
  - aktive Mitarbeit in den Übungen
  - erfolgreiche Abschlussprüfung
- Pool 2.2.07: SWI-Prolog 3.2.8, SML 110.0
- http://www.didaktik.cs.uni-potsdam.de/lehre/

Der Computer

Eine „dumme“ Maschine

zur

Eingabe - Verarbeitung - Ausgabe

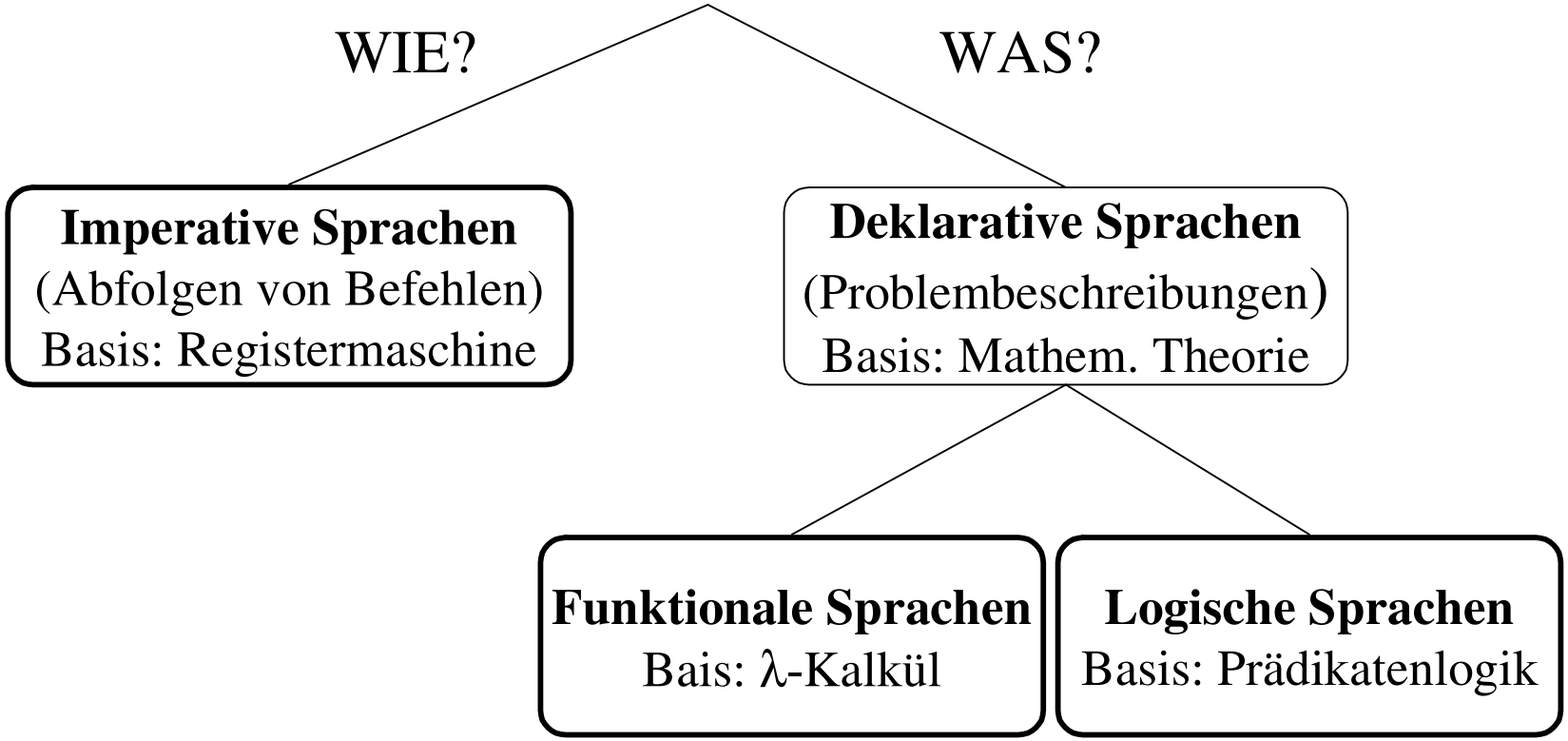
von Zeichen

- Ein-/Ausgabeverhalten veränderbar =>
- Verarbeitungsregeln austauschbar =>

Universalität des Computers

- Formulierung der Verarbeitungsregeln mittels Programmiersprachen
- Welche Zeichenfolgen sind als Programme zugelassen? => Syntax
- Was bewirken diese Zeichenfolgen auf dem Rechner? => Semantik

Welt der Programmiersprachen



Syntax und Semantik von PrgSprachen

Syntax - Lehre vom Satzbau

Semantik - Lehre von der inhaltlichen Bedeutung einer Sprache

- Übersetzersemantik
- operationale Semantik (Interpretersemantik)
- denotationale Semantik (Funktionensemantik)
- axiomatische Semantik (Prädikatensemantik)

- prozedurale Semantik
- deklarative Semantik

Imperative Programmiersprachen

(FORTRAN 1954, BASIC, ALGOL60, PASCAL, C, ADA)*

- Basis: mathematische Modell der Registermaschine
- Programm: Zusammenstellung von Anweisungen (Befehle)
- Programmstart: Ausführung der ersten Anweisung
- Anweisung: Handlungsaufforderung an den Computer
  a) elementare Anweisungen (Bsp. Zuweisungen)
  b) strukturierte Anweisungen (werden durch Konstruktoren aus elem. Anweisungen gebildet: Konatenation, Alternative, Iteration, ...)
- Variable: Behälter mit Bezeichner und Lese-/Schreibop. (Von-Neumann-Architektur)

*Sprachen mit objektorientierter Erweitung: JAVA, C++, DELPHI

Bsp. Pascal

```
program misch(f,g,h);
var f,g,h: file of integer;
begin
    reset(f); reset(g); rewrite(h);
    while not (eof(f) or eof(g)) do
    begin
        if f↑ < g↑ then
        begin
            h↑:=f↑; put(h); get(f)
        end else
        begin
            h↑:=g↑; put(h); get(g)
        end
    end;
    while not eof(f) do
    begin
        h↑:=f↑; put(h); get(f)
    end;
    while not eof(g) do
    begin
        h↑:=g↑; put(h); get(g)
    end
end.
```

Funktionale Programmiersprachen

(LISP 1958, LOGO, FP, ML, Miranda, Haskell)

- Basis: λ-Kalkül
- Programm: Menge von Funktionsdefinitionen
- Programmstart: Aufruf einer parametrisierten Funktion
- Funktion: durch Rechenvorschrift def. Abbildung f: A→B

  a) elementare Funktionen (Bsp. Add, Sub, Operationen auf Listen)

  b) drei zentrale Konstruktoren zur Bildung höherer Funktionen
    - Komposition
    - Anwendung
    - Abstraktion
- Variable: Bezeichner für ein konkretes Objekt

Bsp. ML

```
fun  misch [ ] g  = g: int list  |
     misch f [ ] = f   |
     misch (x::r)(x'::r')=if x<x' then [x]@(misch r (x'::r'))
                                  else [x]@(misch (x::r) r').
```

```
misch [3,7,19,54] [2,3,5,16,74].
```

Logische Programmiersprachen
(PROLOG 1973)

- Basis: Prädikatenlogik 1. Stufe
- Programm: Menge von wahren Aussagen über Sachverhalte (Fakten) + logische Schlussregeln
- Programmstart: Eingabe einer Behauptung (Anfrage)
- Hornklauseln der Prädikatenlogik
  - Fakten: atomare Formeln, d. h. Applikationen von Prädikatssymbolen auf Parameterterme
  - Regeln: Implikationen, deren Rumpf eine Konjunktion von atomaren Formeln ist
- Variable: Unbestimmte

Bsp. Prolog

```
misch([ ],Y,Y).
misch(Y,[ ],Y).
misch([A|X],[B|Y],[A|Z]) :– A≤B, misch(X,[B|Y],Z).
misch([A|X],[B|Y],[B|Z]) :– B<A, misch([A|X],Y,Z).
```

```
?- misch([2,5],[1,2,8],Y).
```

Ausgabe von PROLOG:

```
Y=[1,2,2,5,8].
```